Network Vulnerability Assessment Report

Sample Report — Anonymized

Assessment Date:	April 2026
Target:	[TARGET-IP] / [TARGET-DOMAIN]
Scanner:	OpenVAS / Greenbone Community Edition 26.x
Scan Type:	Full Network Scan (unauthenticated)
Scan Duration:	~55 minutes
Assessor:	Artem Hrechnyi, Security Engineer

1. Executive Summary

This report presents the results of a network vulnerability assessment conducted on a Linux-based server running multiple web services. The assessment was performed using OpenVAS (Greenbone Community Edition) with an unauthenticated full network scan.

The scan identified **2 Low-severity findings** and no Critical, High, or Medium vulnerabilities. The overall security posture of the target is **good**. Both findings are informational in nature and carry minimal exploitability risk. Remediation recommendations are provided below.

2. Scope and Methodology

Target: Single host — [TARGET-IP] ([TARGET-DOMAIN])
Scan type: Unauthenticated full network scan
Tool: OpenVAS Greenbone Community Edition 26.x
Authorization: Performed on infrastructure owned and operated by the assessor

3. Results Overview

Severity	Count	CVSS Range
Critical	0	9.0 – 10.0
High	0	7.0 – 8.9
Medium	0	4.0 – 6.9
Low	2	0.1 – 3.9
Informational / Log	83+	N/A

4. Detailed Findings

4.1 TCP Timestamps Information Disclosure

Severity	Low
CVSS Score	2.6
Port / Protocol	general/tcp
QoD	80%

Description: The remote host implements TCP timestamps (RFC1323/RFC7323). A side effect of this feature is that the approximate uptime of the host can be calculated from the timestamp values, potentially providing an attacker with useful reconnaissance information.

Impact: Low. An attacker may estimate server uptime, which could inform patch cycle assumptions. No direct exploitation path.

Remediation: To disable TCP timestamps on Linux, add the following line to */etc/sysctl.conf* and apply:

```
net.ipv4.tcp_timestamps = 0
sysctl -p
```

4.2 Weak MAC Algorithm(s) Supported (SSH)

Severity	Low
CVSS Score	2.6
Port / Protocol	22/tcp (SSH)
QoD	80%

Description: The SSH server supports weak MAC (Message Authentication Code) algorithms: *umac-64-etm@openssh.com* and *umac-64@openssh.com*. These are 64-bit based algorithms considered cryptographically weak by current standards.

Impact: Low. In theory, a sufficiently resourced attacker could exploit weak MACs to manipulate SSH session integrity. Practical exploitation is unlikely but represents a hardening gap.

Remediation: Disable weak MAC algorithms in */etc/ssh/sshd_config*:

```
MACs
hmac-sha2-256,hmac-sha2-512,hmac-sha2-256-etm@openssh.com,hmac-sha2-512-etm@openssh.com
systemctl restart sshd
```

5. Remediation Roadmap

Priority	Finding	Effort	Timeline
1	Weak SSH MAC Algorithms	Low (config change)	Immediate
2	TCP Timestamps Disclosure	Low (sysctl)	Within 30 days

6. Conclusion

The target host demonstrates a strong security posture with no Critical, High, or Medium vulnerabilities identified. Both Low-severity findings are easily remediated through standard Linux hardening procedures. The server's SSL/TLS configuration supports Perfect Forward Secrecy (PFS) and modern cipher suites, which is a positive indicator of good security practice.

Recommendation: Apply the two remediations above and schedule a follow-up credentialed scan to identify any additional findings that require authenticated access.

Note: This is a sample anonymized report for portfolio purposes. All identifying information (IP addresses, hostnames, domain names) has been replaced with placeholders. Assessor: Artem Hrechnyi | Security Engineer | github.com/leobayker